Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions

Emmanuel Bresson (ENS),
Olivier Chevassut (LBNL-UCL),
David Pointcheval (ENS)

Passer à la première page

OUTLINE

- Motivation and Previous Work
- Communications and Security Model
- A Secure Group DH Protocol
- Standard Assumptions
- Security Theorem and its Proof
- Conclusion

Motivation

- An increasing number of distributed applications need to communicate within groups, e.g.
  - collaboration and videoconferencing tools
  - replicated servers and distributed computations
- An increasing number of applications have security requirements
  - privacy of data
  - protection from hackers, viruses and trojan horses
- Group communication must address security needs

The Problem

- Group Characteristics
  - group relatively small (<100 members), dynamic
  - members have similar computing power
  - no centralized server
- Goals for Group Key Exchange
  - Authenticated Key Exchange (AKE)

    implicit authentication: only the intended partners get *sk*

    semantic security: no information leaks about *sk*
  - Mutual Authentication (MA)

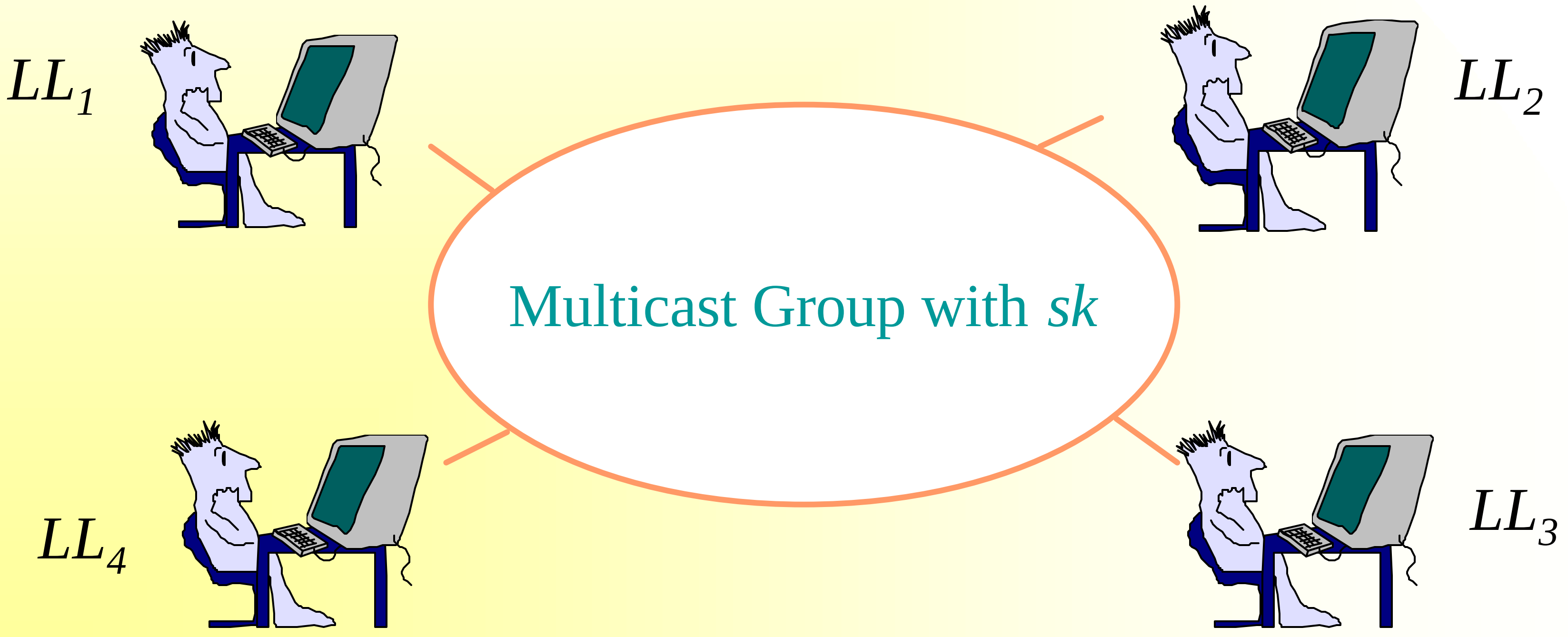
    key confirmation mechanism

Prior Work

- "Provably Authenticated Group DH Key Exchange: The Dynamic Case", [A'01]
  - model of computation in the Bellare-Rogaway style
    - adversary controls the network
    - adversary's interacts with players via oracle queries
  - a group DH key exchange protocol
    - *SETUP, JOIN, REMOVE* algorithms
  - security proof
    - sequential executions only
    - ideal-hash assumption

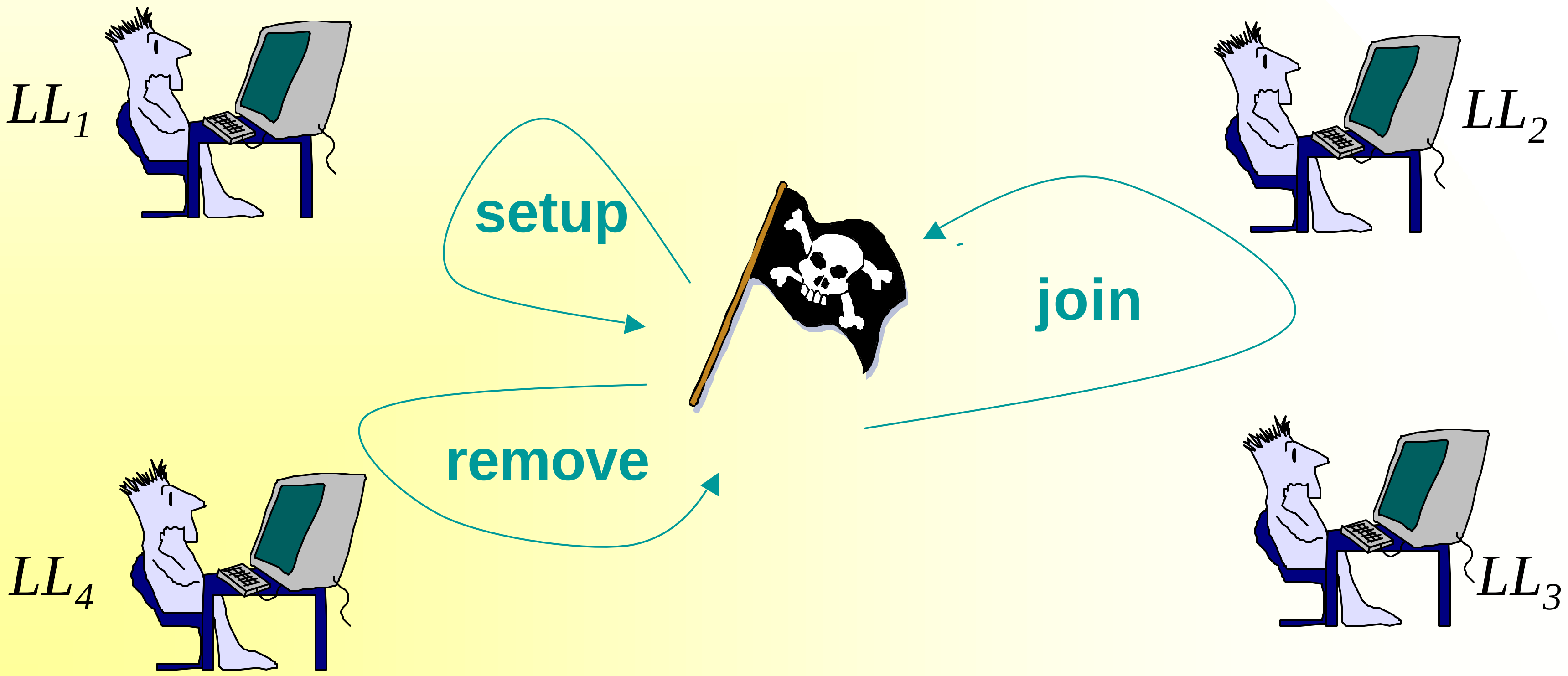
Model of Communication

- A multicast group consisting of a set of players
  - each player holds a long-lived key (LL)
  - each player holds ephemeral internal keying material

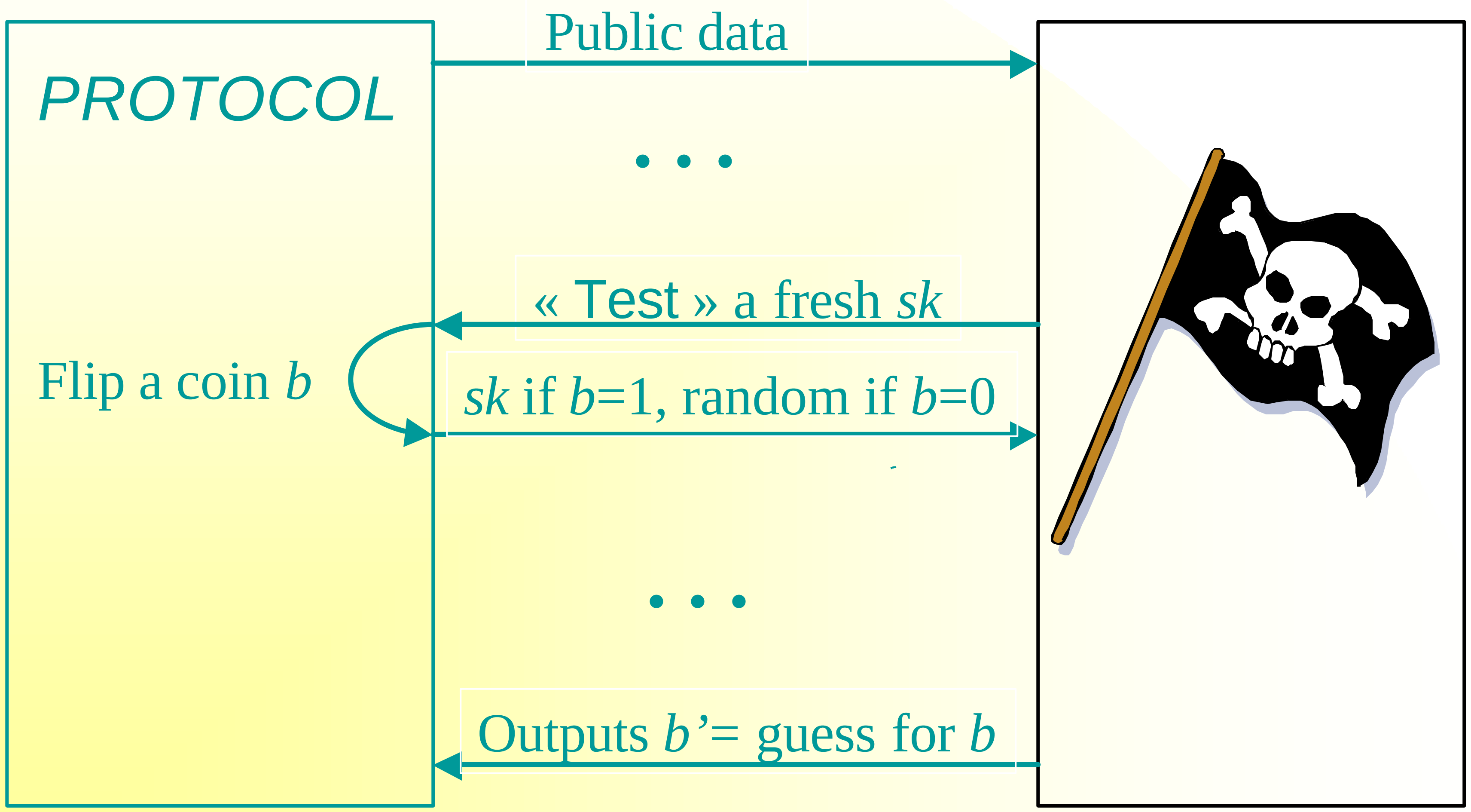


Modeling the Adversary

- Adversary's interacts with the group via queries
  - setup: initialize the multicast group
  - join/remove: add or remove players to multicast group



Security Definitions (AKE)



OUR CONTRIBUTIONS

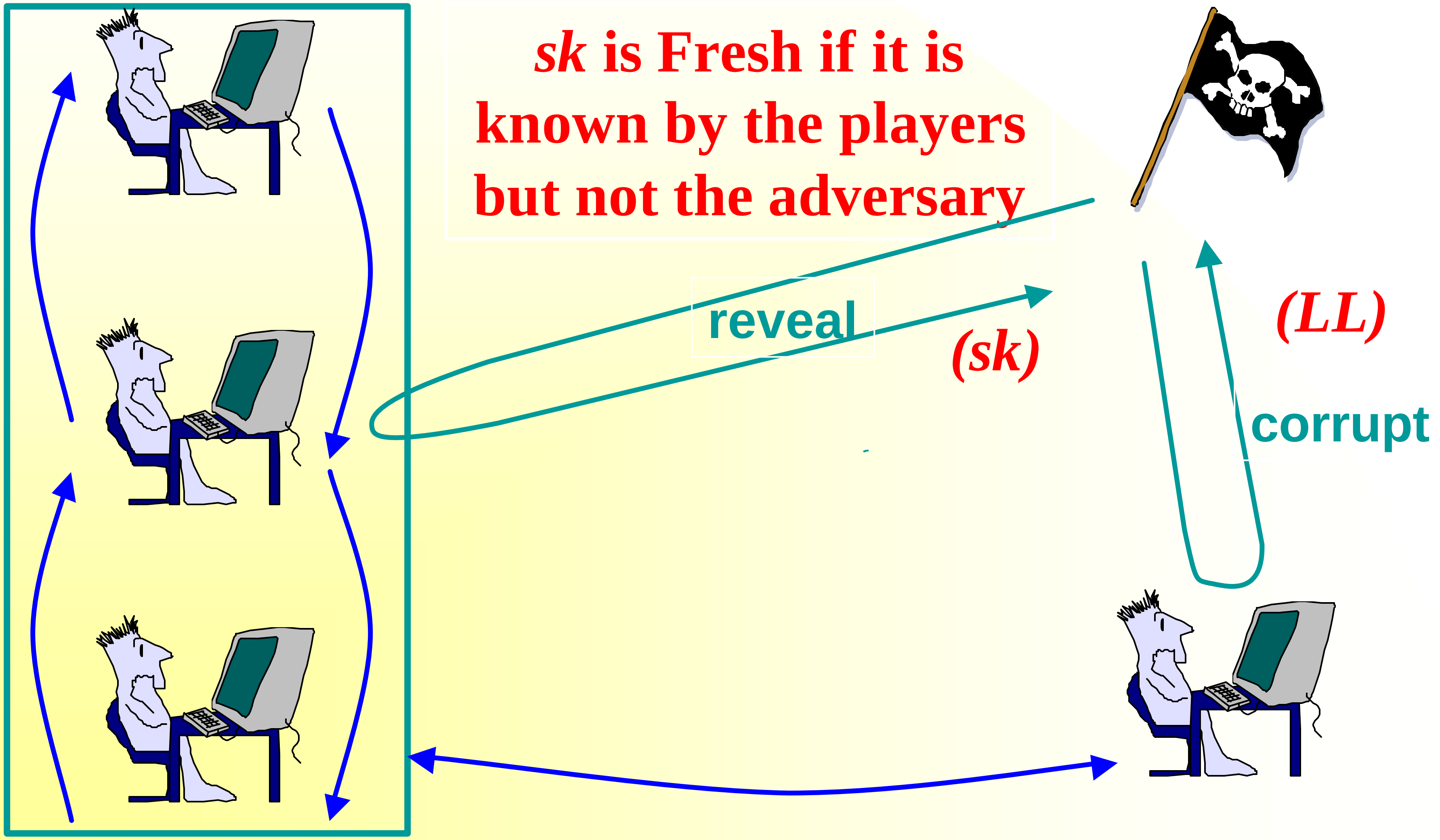
- Concurrent executions considered
- Forward-secrecy
  - Strong-corruption and weak corruption
- Use of secure crypto-devices
  - Crypto-processor and smart card
- Standard assumptions
  - No random oracle
  - Improved security proof

Forward-Secrecy

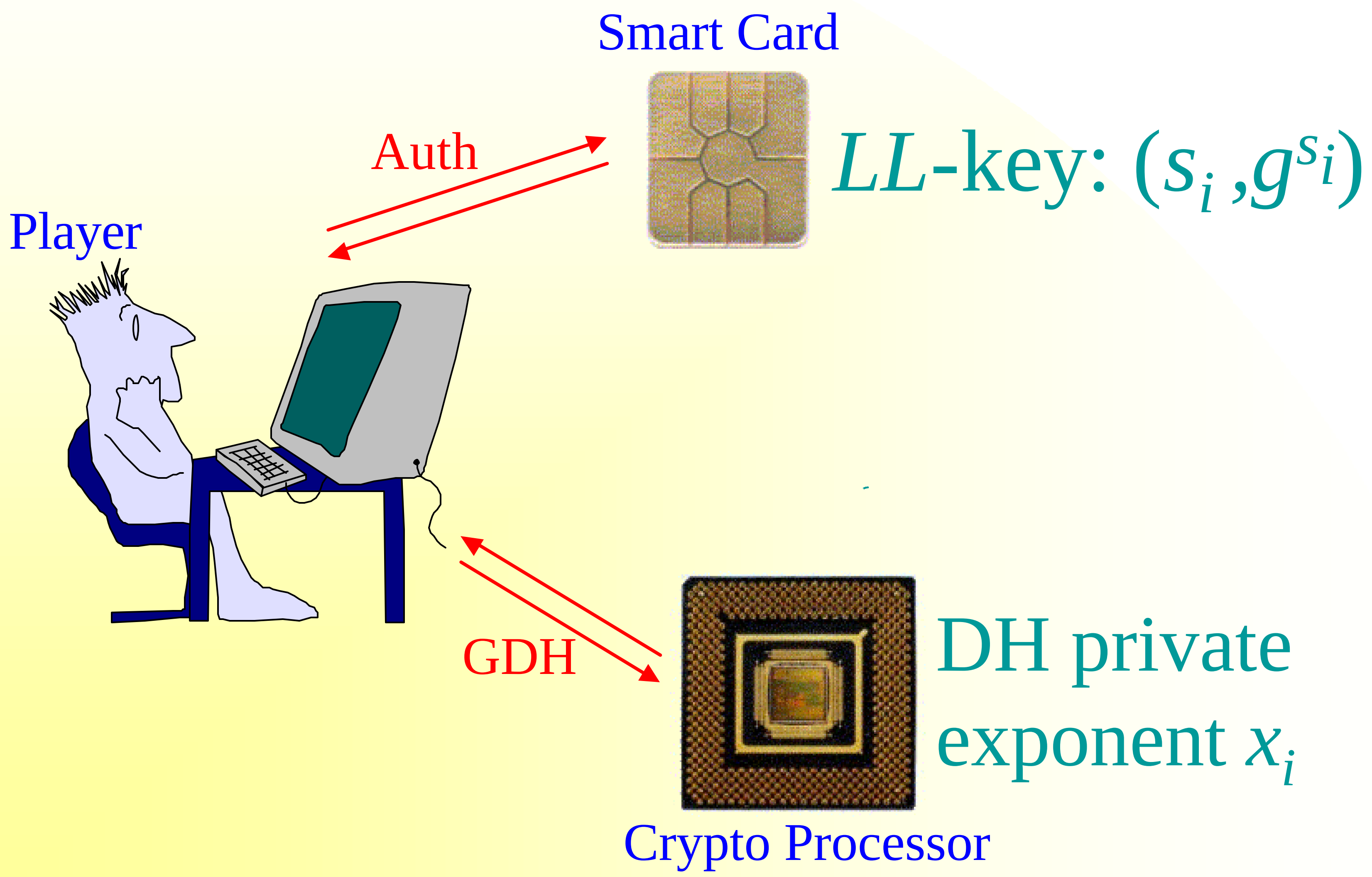
- The corruption of long-lived keys LL should not entail the security of *previously* established session keys sk.
- 2 flavors of forward-secrecy can be defined:
  - Weak-corruption model: adversary can obtain LL-keys only.
  - Strong corruption model: adversary may corrupt private exponents as well.

Freshness vs. Corruption Queries

sk is Fresh if it is known by the players but not the adversary



Crypto-Devices

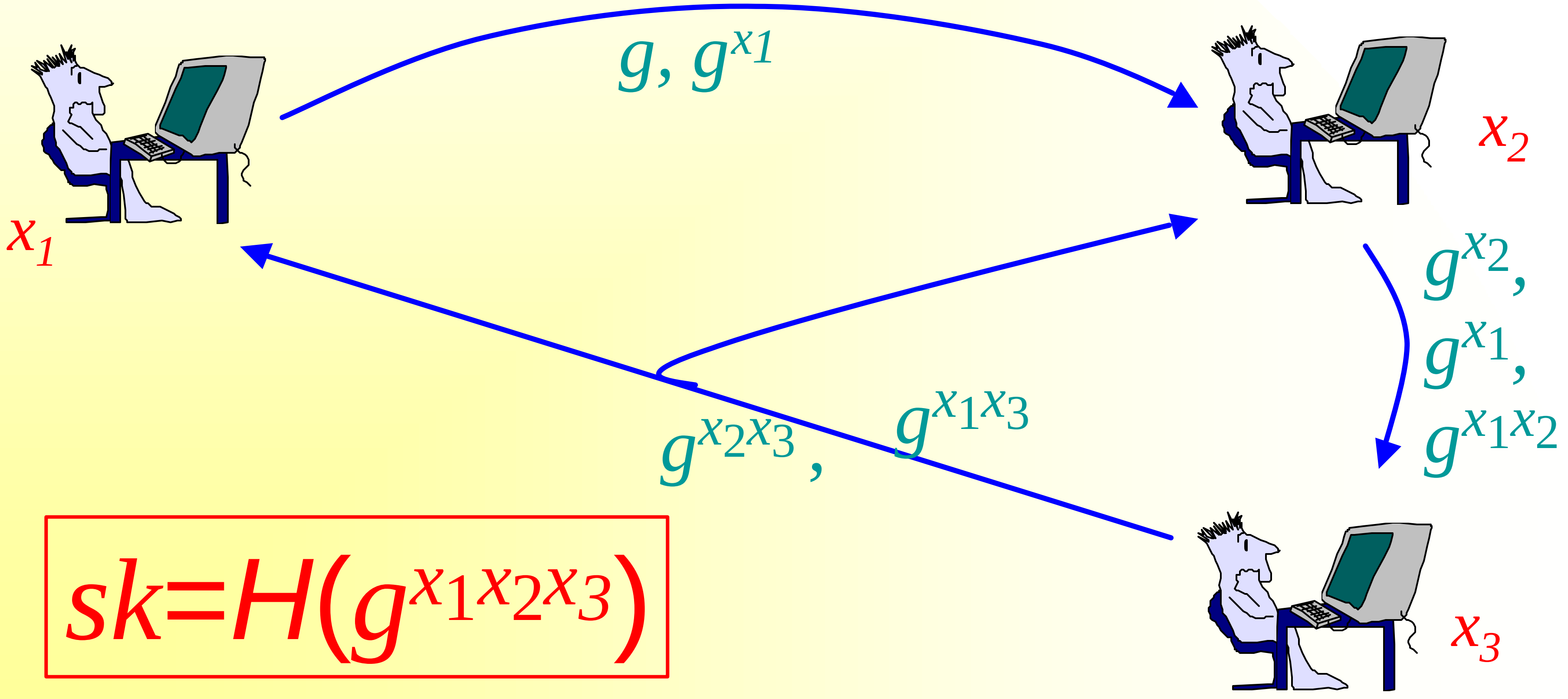


A Secure Authenticated Group Diffie-Hellman Protocol

- The session key is
  - $sk = H(g^{x_1 x_2 \cdots x_n})$
- Ring-Based with flows
- Defined by three algorithms
  - *SETUP*
  - *REMOVE*
  - *JOIN*
- Many details abstracted out

The *SETUP* Algorithm

- Up-flow: U_i raises to x_i and forwards to U_{i+1}
- Down-flow: U_n raises to x_n and broadcasts

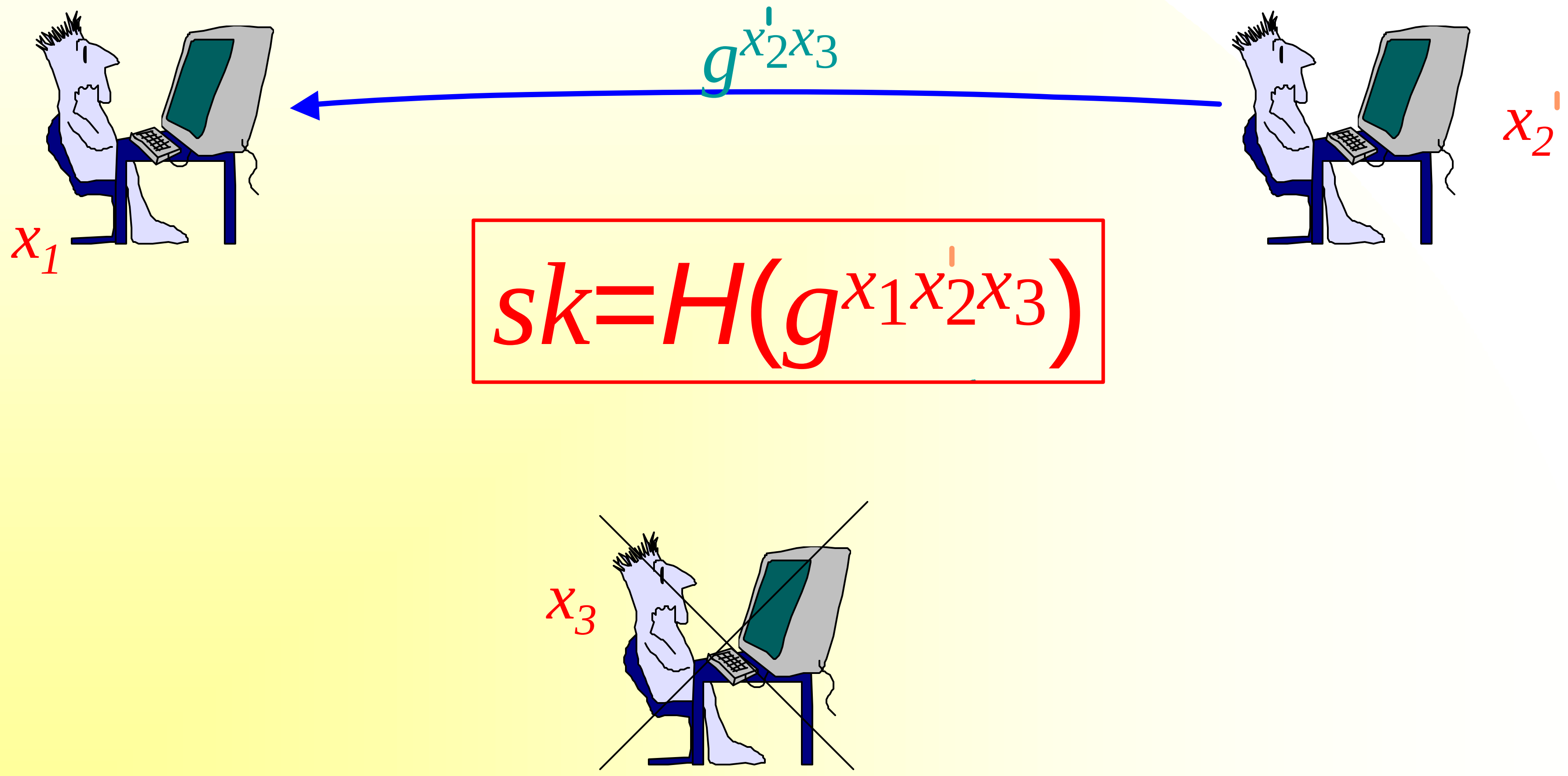


x_1
g, g^{x_1}
x_2
$g^{x_2}, g^{x_1}, g^{x_1 x_2}$
$g^{x_2 x_3}, \quad g^{x_1 x_3}$
x_3

$$\boxed{sk = H(g^{x_1 x_2 x_3})}$$

The *REMOVE* Algorithm

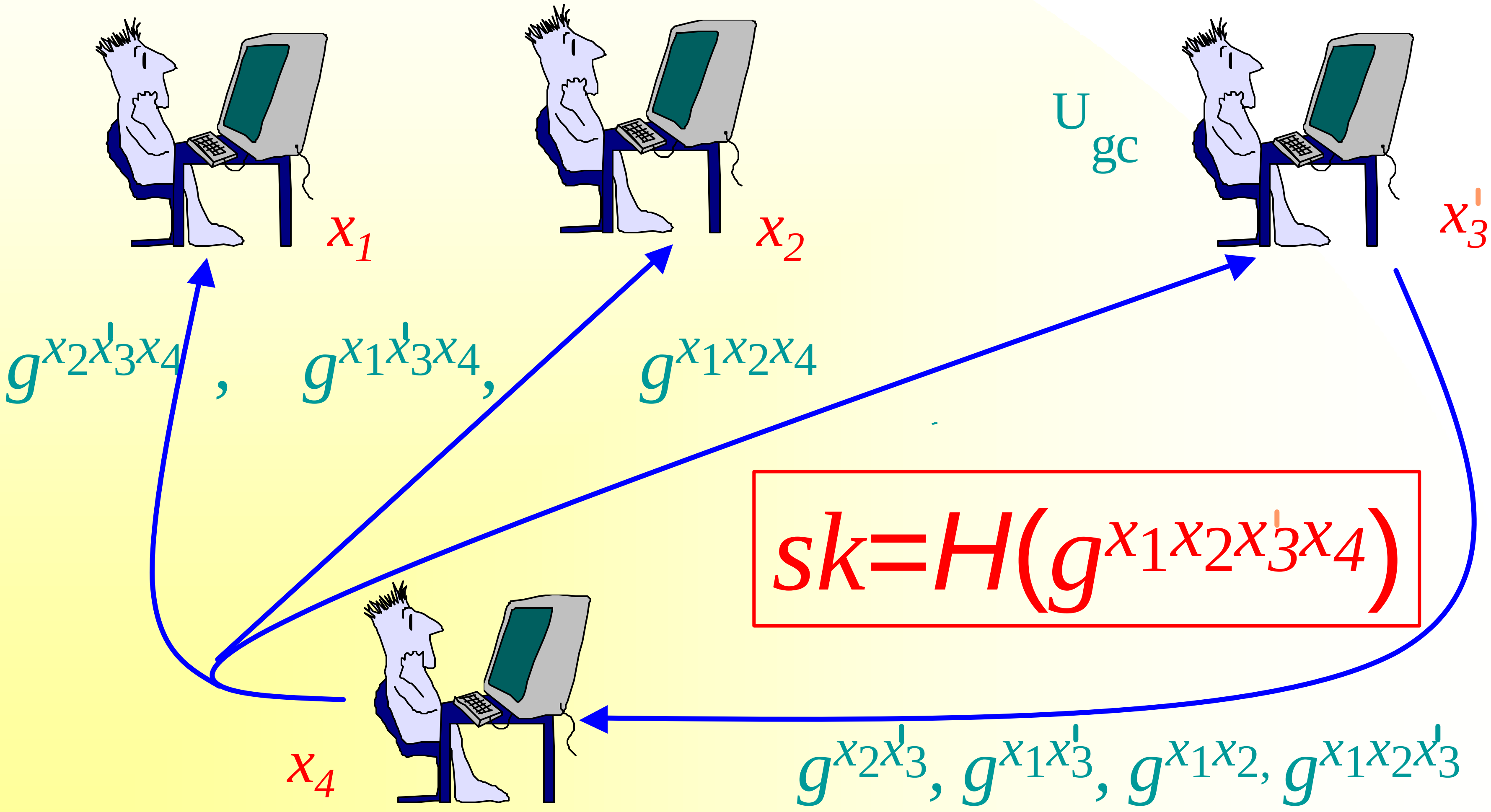
- Down-flow of the SETUP algorithm



$$\boxed{sk=H(g^{x_1 x_2' x_3})}$$

The *JOIN* Algorithm

- SETUP initiated by player with highest index in group



x_1
x_2
U_{gc}
x_3'
x_4
$g^{x_2 x_3' x_4}$, $g^{x_1 x_3' x_4}$, $g^{x_1 x_2 x_4}$
$sk=H(g^{x_1 x_2 x_3' x_4})$
$g^{x_2 x_3'}$, $g^{x_1 x_3'}$, $g^{x_1 x_2}$, $g^{x_1 x_2 x_3'}$

Standard assumptions

- Group Decisional Diffie-Hellman
- Multi-Decisional Diffie-Hellman
- Message Authentication Codes (MAC)
- Entropy-smoothing theorem

GDDH Assumptions

- Given some subsets of indices in $I=\{1,\ldots,n\}$, and all values:

  $g^{?_{i ? J} x_i}$, for every given subset J
- Decide whether a given value is

  $g^{x_1 \cdot \ldots \cdot x_n}$ or not.
  - Eg.: Given g^a, g^b, g^{ac}, distinguish g^{abc} from a random value g^r.

Multi-DDH Assumptions

- Given some n values: g^{x_i}, for $i=1,\ldots n$
- Decide whether $n(n-1)/2$ values are

  $g^{x_i x_j}$ or not.

  - Eg.: Given g^a, g^b, g^c, distinguish g^{ab}, g^{ac}, g^{bc} from three random values g^r, g^s, g^t.

- MDDH problem can easily be reduced to DDH

Message Authentication Codes (MAC)

- Kgen: outputs a random string k of length l
- Sign/Verif: produces and verifies a MAC from m and k

- MACs will be used to authenticate the flows between players

- MACs exist if OW-functions exist.

Entropy-smoothing theorem

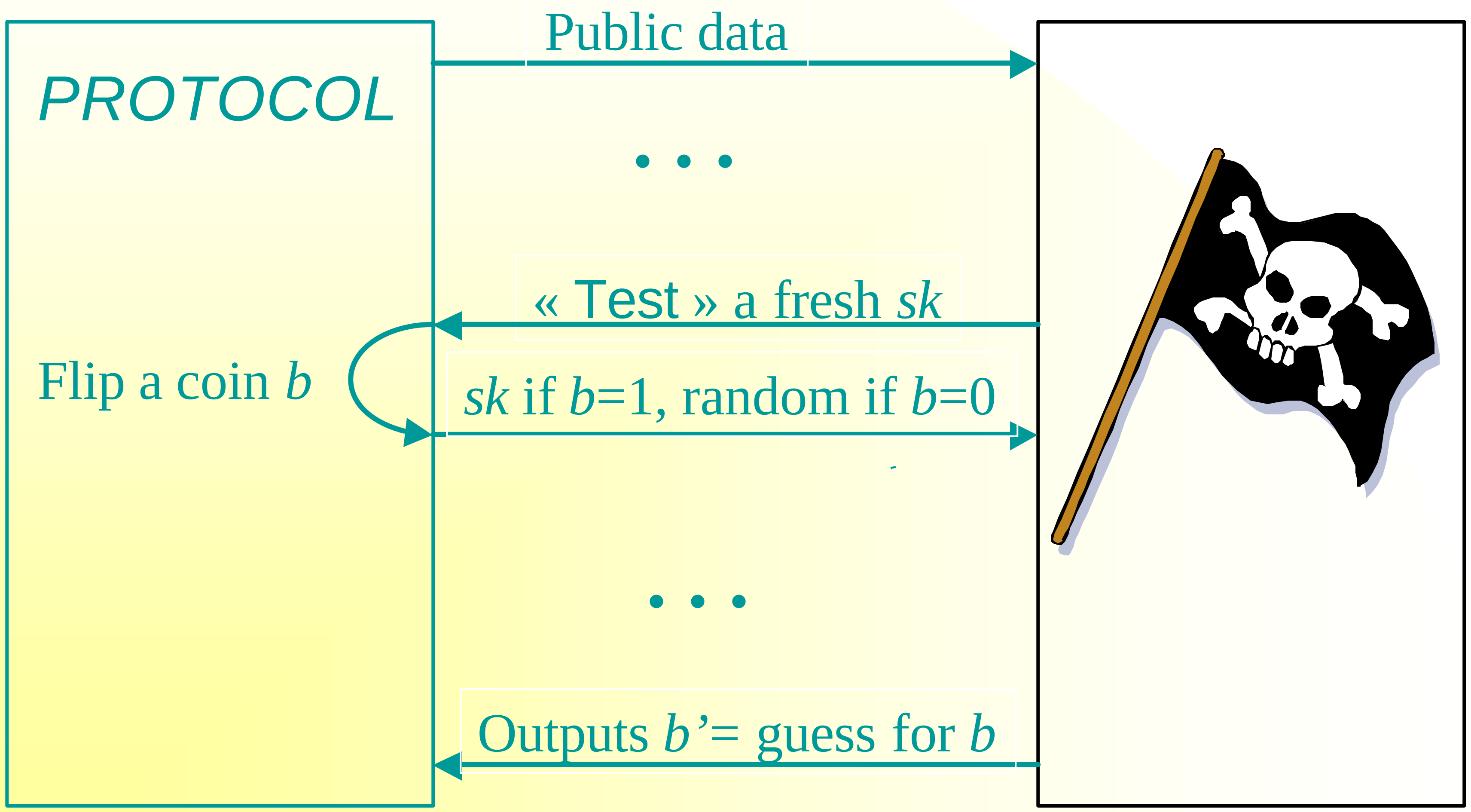
- Used to derivate keys from $g^{\cdots}$
- Let D be a distribution of length s and entropy $?$. Let H be a universal hash function from k-bits x s-bits to l-bits.

- Then the following $(l+k)$-bits distributions are $2^{-(e+1)}$-statistically close, where $l=?-2e$:

$$H_r(x)||r \qquad \text{and} \qquad y||r$$

$x?_D\{0,1\}^s$, Uniform

Security Definitions (AKE)



Security Theorem (AKE)

- Security is measured as the adversary's advantage in guessing the bit *b* involved in the Test-query
- This advantage is a function of
  - the adversary's advantage in breaking the Group DDH
  - the adversary's advantage in breaking the MAC scheme
  - the adversary's advantage in breaking the Multi-DDH
- Theorem

$$\mathrm{Adv}^{\mathrm{ake}}(T,Q,q_s) \ ?\ 2nQ\cdot\mathrm{Adv}^{\mathrm{gddh}}(T') + n(n-1)\cdot\mathrm{Succ}^{\mathrm{cma}}(T)$$
$$+ 2\cdot\mathrm{Adv}^{\mathrm{mddh}}(T') + \text{« negligible terms »}$$
$$T' ?\ T + nQ\cdot T_{\mathrm{exp}}(k)$$

Sketch of Proof

Game 0: Real attack

Game 1: Abort if a MAC forgery occurs

Game 2: Guess the execution in which the Test-query occurs

Game 3: Simulate the flows from a true GDDH tuple based on the guesses above

Game 4: Simulate the flows, but with a bad GDDH tuple

Game 5: Answer the Test-query at random, letting no advantage to the adversary.

Difference with [A01]

- Group Diffie-Hellman term is relative to the total number of players n (instead of the size of multicast group s)
  - Loss from $O(s^3)$ $\mathrm{Adv}^{\mathrm{ddh}}$ to $O(n^3)\mathrm{Adv}^{\mathrm{ddh}}$
- But :
  - Improved by a binomial factor from $s\binom{n}{s}$ to n
  - Better compared to n^3/s^3
- ? This is a good deal

Conclusion and Future Work

- Summary
  - a model for strong-corruption
  - a proof allowing for concurrent sessions
  - a proof that does not require an ideal-hash assumption
- Work in Progress
  - “Group DH Key Exchange secure Against Dictionary Attacks”